

REMARKS

In this response, claims 1-6, 8-15, and 17-23 are pending. No claim is amended.

The final Office Action mailed January 15, 2004 rejected claims 1-6, 8-15, and 17-23 under 35 U.S.C. § 102 as anticipated by *Biliris et al.* (US 5,590,327). This rejection is respectfully traversed for the following reasons.

Claims 19-23 are patentable over *Biliris et al.* because *Biliris et al.* does not show the recited “padding element.” The final Office Action contends that this feature is shown in col. 4:51-64 as follows:

As can be seen from the illustrations, each object of a class that has virtual functions contains a hidden pointer that points to a virtual function table, called the vtbl. The vtbl contains the addresses of the specific virtual functions to be called for the given object. In the case of derived class objects, the vtbl of a base class sub-object also contains offsets (deltas) that are used to find the address of the derived class object given the address of the base class sub-object.

In FIG. 2, vtbl pointer **22** is the first entry in the memory layout of person object **21**. Also included in the memory layout are, of course, the entries for the data members of the class, firstname **23**, lastname **24** and age **25**.

This portion of *Biliris et al.* is silent about any padding element at all. The vtbl pointer **22** is not padding but a useful pointer used by all instances of classes with virtual functions. In any event, vtbl pointer **22** fails to satisfy other recitations such as “the first layout for the slot of the object in the high-order language includes a padding element and the second layout for the slot of the object in the high-order language does not include the padding element” in independent claim 21.

Biliris et al. also does not disclose the limitations of claims 1-6 and 10-15. For example, independent claims 1 and 10 recite: “generating a layout for the object in a high-order language based on the definition of the object and the size and **alignment of the one or more primitive types.**” The generation of object layouts based on the size and alignment advantageously results

in an object system for use in a run-time environment that is both portable and maintainable (see Spec. p. 7).

This feature, however, is not shown in *Biliris et al.* In fact, *Biliris et al.* is unconcerned with either the problems with or the details of generating object layouts, and the portions cited in *Biliris et al.* do not go beyond the ordinary use of C++. For example, col. 4:30-50 shows a class definition including an array defined as `char firstname[MAX]` for which the task of the C++ compiler is to “generate code” (col. 1:61-63), but this definition does not define an “alignment of one or more primitive types” as recited in claims 1 and 10. The final Office Action’s response to arguments, “*Biliris* teaches the defied [*sic*, defined?] data (lines 30-49 column 4) included size and alignment (`char [MAX]`),” does not support the rejection, because `char` is already a type and `char [MAX]` does not define the “alignment” of “char” objects. In fact, there is no way to specify the alignment of types in standard, portable C++, nor does *Biliris et al.* show any non-standard specification of alignment of types.

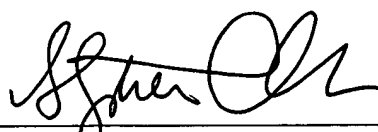
As for the rejection of claim 8-9 and 17-18, that too is respectively traversed. Claims 8 and 9 recite “generating a plurality of layouts, corresponding respectively to the incompatible platforms,” but *Biliris et al.* does not even disclose multiple platforms, not to mention whether the platforms are incompatible. FIG. 3, referenced in the Office Action, shows a memory layout for a student object **31** that includes a student subobject **58** and a person subobject **36**. However, there is no disclosure that the memory space for student object **31** exists on a plurality of incompatible platforms. As for claim 9, col. 6:52-59 merely discusses the use of a hidden pointer to the virtual function table with no mention of “incompatible platforms,” and col. 7:17-30 relates only to the use of different virtual function tables for different sub-objects in situations of multiple inheritance, again without disclosure of “incompatible platforms,” as recited in claims 8-9 and 17-19.

Therefore, the present application, as amended, overcomes the objections and rejections of record and is in condition for allowance. Favorable consideration is respectfully requested. If any unresolved issues remain, it is respectfully requested that the Examiner telephone the undersigned attorney at 703-425-8516 so that such issues may be resolved as expeditiously as possible.

Respectfully Submitted,

DITTHAVONG & CARLSON, P.C.

8/16/04
Date



Stephen C. Carlson
Attorney/Agent for Applicant(s)
Reg. No. 39929

10507 Braddock Rd
Suite A
Fairfax, VA 22032
Tel. 703-425-8516
Fax. 703-425-8518